CLAIMS

What is claimed is:

1. A processor, comprising:

a multi-entry stack usable in at least a stack-based instruction set;

logic coupled to said stack, the logic manages the stack; and

a plurality of registers coupled to the logic and addressable through a second instruction set that provides register-based and memory-based operations.

2. The processor of claim 1 wherein the stack has a top and the stack is accessible within the second instruction set through at least one of the registers in which a value is stored that is present at the top of the stack.

3. The processor of claim 1 wherein the stack has a top that is addressable by a memory mapped address, and the memory mapped address is stored in a register which is accessed by the second instruction set.

4. The processor of claim 1 wherein the stack-based instruction set accesses operands from the stack and places results from operations on the stack and, as a result of accessing operands from the stack and placing results on the stack, at least some of the registers are updated.

5. The processor of claim 1 further comprising a first program counter usable in the execution of the stack-based instruction set and a second program counter usable in the execution

of a micro-sequence that comprises instructions from both the stack-based and second instruction sets.

6. The processor of claim 1 further comprising a pair of parallel address generation units coupled to said logic which are used to compute memory source and destination addresses and wherein a register includes the top of the multi-entry stack, thereby permitting a block of data to be moved between a memory area and the stack by execution of a single instruction with a repeat loop.

7. The processor of claim 1 wherein the second instruction set comprises an instruction that retrieves operands from memory, performs a computation on the operands, and places the result on the stack.

8. A method of processing instructions in a processor, comprising:

fetching instructions from a first instruction set which comprises stack-based instructions;

fetching instructions from a second instruction set which comprises memory-based and register-based instructions; and

executing said fetched instructions from the first and second instruction sets.

9. The method of claim 8 further comprising forming a sequence of instructions from both of said first and second instruction sets.

10. The method of claim 8 further comprising executing an instruction from said second instruction set that targets a stack included in said processor, said stack having a top, and storing a value at the top of the stack in a register in the processor.

11. The method of claim 10 further comprising updating an address stored in another register that points to the top of the stack.

12. A processor, comprising:

a multi-entry stack having a top and usable in at least a stack-based instruction set;

logic coupled to said stack, the logic manages the stack;

memory coupled to said logic; and

a plurality of registers coupled to the logic and addressable through a second instruction set that provides register-based and memory-based operations;

wherein a first register includes an address through which the top of the stack is accessed and a second register in which a value at the top of the stack is stored.

13. The processor of claim 12 wherein the stack-based instruction set accesses operands from the stack and places results from operations on the stack and, as a result of accessing operands from the stack and placing results on the stack thereby causing the address in the first register to be changed.

14. The processor of claim 13 wherein the address in the first register is incremented or decremented depending on whether the register is used as a source or a destination, respectively, for an operation.

15. The processor of claim 12 wherein the stack-based instruction set comprises Java Bytecodes.

16. The processor of claim 12 further comprising a first program counter usable in the execution of the first instruction set and a second program counter usable in the execution of code that comprises instructions from both the first and second instruction sets.

17. The processor of claim 12 wherein at least one of the registers are used to calculate addresses in parallel, said addresses being calculated in accordance with any of a plurality of addressing modes specified by the second instruction set.

18. The processor of claim 17 wherein at least one of the registers includes an offset usable in the calculation of addresses.

19. The processor of claim 12 wherein the second instruction set comprises an instruction that moves data from a register or memory to a register, and consequently to the stack.

20. The processor of claim 19 wherein the instruction that moves data includes a plurality of bits of that encode one of a plurality of addressing modes.

21. The processor of claim 20 wherein the addressing modes include a mode in which the instruction that moves data includes an immediate value and a reference to a register containing a base address, wherein the immediate value and the base address are added together to generate a source memory address for the move instruction.

22. The processor of claim 20 wherein the addressing modes include a mode in which the instruction that moves data includes a reference to register in which a source memory address is stored to be used in the move instruction, and the source memory address in the referenced register is incremented by an immediate value also included in the move instruction.

23. The processor of claim 20 wherein the addressing modes include a mode in which the instruction that moves data includes references to two registers in which memory addresses are stored, one register being a predetermined index register, the memory addresses from the two registers are added together to calculate a source memory address used to complete the move instruction, and the address in the predetermined index register is incremented.

24. The processor of claim 20 wherein the addressing modes include a mode in which the instruction that moves data includes references to two registers in which memory addresses are stored, the memory addresses are added together to calculate the memory address used to complete the move instruction.

25. The processor of claim 12 wherein the processor is configured to be coupled to a separate processor on which an operating system is executed.

26. The processor of claim 12 further comprising a first program counter usable in the execution of the stack-based instruction set and a second program counter usable in the execution of a micro-sequence that comprises instructions from both the stack-based and second instruction sets.

27. The processor of claim 12 further comprising a pair of parallel address generation units coupled to said logic which are used to compute memory source and destination addresses and wherein a register includes the top of the multi-entry stack, thereby permitting a block of data to be moved between a memory area and the stack by execution of a single instruction with a repeat loop.

28. A processor, comprising:

a multi-entry stack;

memory;

logic coupled to said stack and said memory;

a plurality of registers coupled to the logic; and

a means for fetching and processing instructions from at least two instruction sets including a stack-based instruction set and a second instruction set that provides register-based and memory-based operations.

29. The processor of claim 28 wherein the stack includes a top and a register includes a pointer to the top of the stack and another register includes a data value stored at the top of the stack.

30. A system, comprising:

a main processor unit;

a co-processor comprising a stack and registers, said co-processor is coupled to the main processor unit and the co-processor is configured to execute stack-based instructions from a first instruction set and instructions from a second instruction set that provides memory-based and register-based operations.

31. The system of claim 30 wherein the stack-based instructions comprise Java bytecodes.

32. The system of claim 31 further including a compiler coupled to said co-processor, said compiler receives Java bytecodes and replaces at least one group of bytecodes by a sequence of instructions from the second instruction set and provides said sequence to the co-processor for execution.

33. The system of claim 32 wherein the sequence also includes stack-based instructions from the first instruction set.

34. The system of claim 30 wherein the system comprises a cellular telephone.

35. The system of claim 30 wherein the stack has a top and is accessible within the second instruction set through at least one of the registers in which a value is stored that is present at the top of the stack.

36. The system of claim 30 wherein the top of the stack is addressable by a memory mapped address, and the memory mapped address is stored in a register which is accessed by the second instruction set.

37. The system of claim 30 wherein the stack-based instruction set accesses operands from the stack and places results from operations on the stack and, as a result of accessing operands from the stack and placing results on the stack, at least some of the registers are updated.

38. The system of claim 30 further comprising a first program counter usable in the execution of the stack-based instruction set and a second program counter usable in the execution of a micro-sequence that comprises instructions from both the stack-based and second instruction sets.

39. The system of claim 38 wherein the first and second program counters are stored in said registers.

40. The system of claim 30 wherein the co-processor further comprises a first program counter usable in the execution of the first instruction set and a second program counter usable in

the execution of a micro-sequence that comprises instructions from both the first and second instruction sets.

41. The system of claim 30 further comprising a memory area and wherein the co-processor further comprises a pair of parallel address generation units coupled to said logic which are used to compute memory source and destination addresses and wherein a register includes the top of the stack, thereby permitting a block of data to be moved between a memory area and the stack by execution of a single instruction with a repeat loop.